FOSSpicks

Sparkling gems and new releases from the world of Free and Open Source Software

Hunting snarks is for amateurs – **Ben Everard** spends his time in the long grass, stalking the hottest, free-est Linux software around.

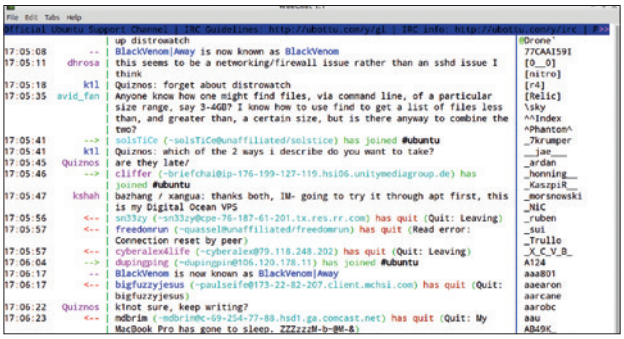
IRC Client

Weechat

IRC is so old it predates the web, yet it's still extremely popular because it works well, and it's widely supported by open source software. This means that anyone can quickly set up a server and have their own group chat. Alternatively, there are many public servers that you can create or join channels on. Almost every free software project has an IRC channel for developers and users to discuss the software. It's also the method of choice we use at Linux Voice for keeping in contact with each other while working on the magazine.

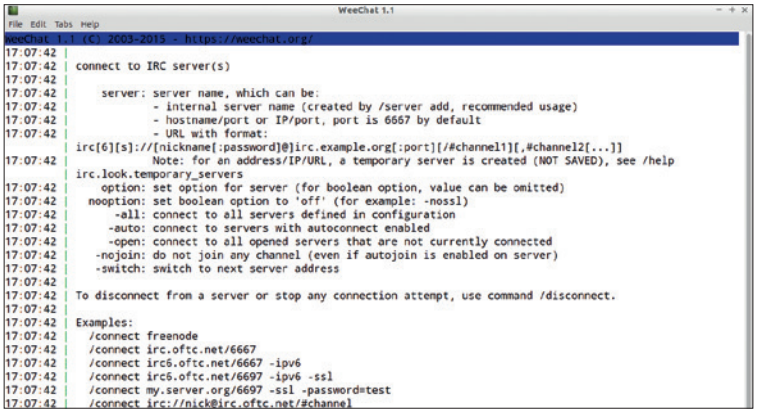
Weechat is a terminal-based client for IRC. In this age of graphical applications, a terminal-based client might sound old-fashioned, but actually, since IRC was designed as a text-based protocol in a world where graphical tools were a rare novelty, the protocol works well in a terminal, and many of the most widely used IRC clients are terminal-based.

IRC servers don't save histories of chats. That means that if you're client is not logged in to a server, then you can't go back and see what was said. This isn't always ideal. One solution many people use is to run a terminal-based IRC client on a server, then SSH into the server to access IRC. This works provided you have an SSH client on every machine you want to access IRC on. This is certainly possible given that even mobile phones can use SSH, but it's not always the most convenient option on mobile devices. *Weechat* can also act as a relay enabling you to connect another client to your *Weechat* session. For example, **www.glowing-bear.org** is a website that can connect to a *Weechat* client and give you access to your *Weechat* session from any machine with a web browser.



Freenode is one of the most popular IRC servers, and most major open source projects have a channel, including us (#linuxvoice).

"Almost every free software project has an IRC channel for developers and users to discuss the software."


WeeChat 1.1
File Edit Tabs Help
WeeChat 1.1 (C) 2003-2015 - https://weechat.org/
17:07:42 | connect to IRC server(s)
17:07:42 |
17:07:42 | server: server name, which can be:
17:07:42 | - internal server name (created by /server add, recommended usage)
17:07:42 | - hostname/port or IP/port, port is 6667 by default
17:07:42 | - URL with format:
| irc[6][s]://[nickname[:password]@]irc.example.org[:port][/#channel1][,#channel2[...]]
17:07:42 | Note: for an address/IP/URL, a temporary server is created (NOT SAVED), see /help
| irc.look.temporary_servers
17:07:42 | option: set option for server (for boolean option, value can be omitted)
17:07:42 | nooption: set boolean option to 'off' (for example: -nossl)
17:07:42 | -all: connect to all servers defined in configuration
17:07:42 | -auto: connect to servers with autoconnect enabled
17:07:42 | -open: connect to all opened servers that are not currently connected
17:07:42 | -nojoin: do not join any channel (even if autojoin is enabled on server)
17:07:42 | -switch: switch to next server address
17:07:42 |
17:07:42 | To disconnect from a server or stop any connection attempt, use command /disconnect.
17:07:42 |
17:07:42 | Examples:
17:07:42 | /connect freenode
17:07:42 | /connect irc.oftc.net/6667
17:07:42 | /connect irc6.oftc.net/6667 -ipv6
17:07:42 | /connect irc6.oftc.net/6697 -ipv6 -ssl
17:07:42 | /connect my.server.org/6697 -ssl -password=test
17:07:42 | /connect irc://nick@irc.oftc.net/#channel

As well as the various user guides, there's also built-in help that can be accessed with **/help <command>**.

There's comprehensive documentation in English, French, Japanese, German, Italian and Polish, so you should be able to get started without any problems. You'll find them at **https://weechat.org/doc/stable**.

One feature of IRC that's missing from most other chat protocols is the support for scripting, and to this end *Weechat* supports six scripting languages, so most programmers should find at least one they're familiar with. There's a repository of scripts at **https://weechat.org/scripts** that you can use if you don't want to write your own.

Although we have no figures to support this, we get the impression that *Weechat* is the fastest growing IRC client in terms of users. In the last couple of years, it's become widely talked about, and rightly so. It's an excellent client.

PROJECT WEBSITE
https://weechat.org/

Pretend computer manipulator

Virtual Machine Manager

There's a wide variety of virtualisation software available. *VirtualBox* is great because the graphical interface makes it easy to learn to use. *Qemu* and *KVM* are really powerful, but it can take time to learn to use the command line tools. Linux containers (LXC) aren't true virtualisation, but can also be really useful if you know how to use them. Wouldn't it be great if there were a tool that brought *VirtualBox*'s ease of use to the power of *Qemu*, *KVM*, LXC and others? Well there is, and it's called *virt-manager* (or *Virtual Machine Manager*).

Red Hat developed *virt-manager* to help users easily take advantage of the full range of virtualisation options available on their enterprise distro, but it's open source and runs on most other distros as well. However, installing *virt-manager* can be a bit tricky because you have to make sure all the requisite components are installed, and that you have the right user permissions. You should look for distro-specific information before getting started.

Once up and running, *virt-manager* enables you to create new machines, edit their hardware, start and stop them, and all the other features you'd expect from a virtual machine manager. You can also use the specific options for the different types of virtual machine. For example, if you use *Qemu*, you can select from any of the architectures your install supports.

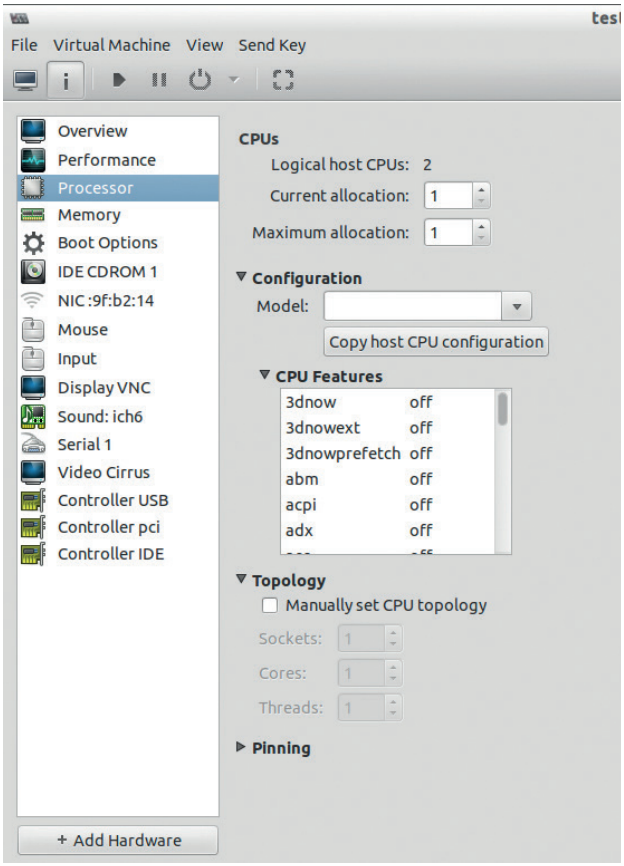
Once your machines are running, you can connect a display and use their graphical output, or connect to their serial console to see what's going on.

Perhaps the biggest disappointment with *virt-manager* is the documentation. Its website only contains a single piece, the FAQ, and this just contains three trivial questions. There's a bit more information on the man page, but this doesn't help much as it's a GUI application. Frankly, we expect a little better from Red Hat. Still, it's fairly straightforward to use, and most people should be able to get started without any problems.

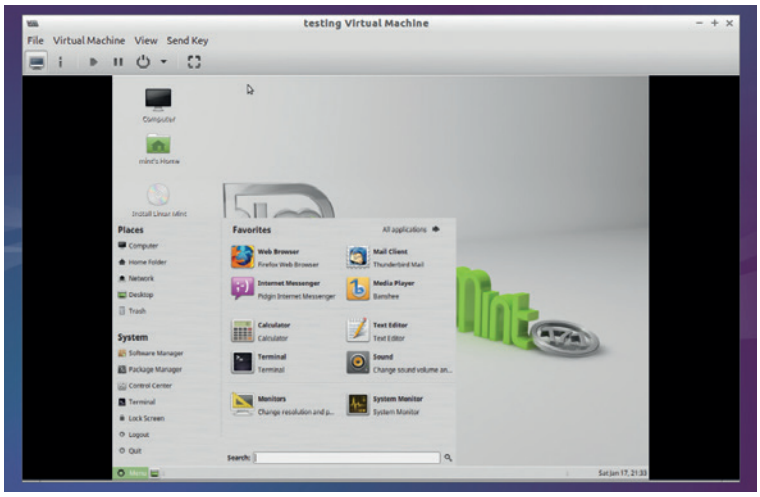
Management layer

Virt-manager is in reality just a graphical layer that sits on top of *libvirt*. It's this library that manages the communication between all the different hypervisors. Since it doesn't actually run the virtual machines (it's just used to manage them), it can distance itself from the virtual machines to a certain degree. For example, you can use *virt-manager* to start a virtual machine, then close *virt-manager* and the virtual machine will continue running. You can then reopen *virt-manager* and use it to shut down the VM. This architecture also means that you can connect it to virtual machines running on different physical computers, provided you have the appropriate permissions. The machines don't have to have been created by virt-manager: as long as they're running on one of the supported platforms, then you can connect to them.

While *virt-manager* is great for setting up machines and getting them running, it's not so great for managing whole groups of them. There are whole other toolsets designed for just this task (such as the ones that come with CoreOS). But for simple use, especially desktop use, *virt-manager* is an excellent tool for taking the pain out of virtual machines.



The huge range of configuration options in *virt-manager* allows you to tailor almost every aspect of your virtual machines.



Virtualisation allows you to try out many distros without rebooting your machine.

PROJECT WEBSITE
https://virt-manager.org/

Python 3D library

Soya 3D

Most modern computers – and even most phones – come with some form of 3D graphical acceleration. This hardware enables the computer to render incredibly complex scenes in real time. OpenGL is the standard rendering library for Linux, and it's very powerful, but it's also quite low-level and requires the programmer to use vector-matrix algebra to manipulate the scene.

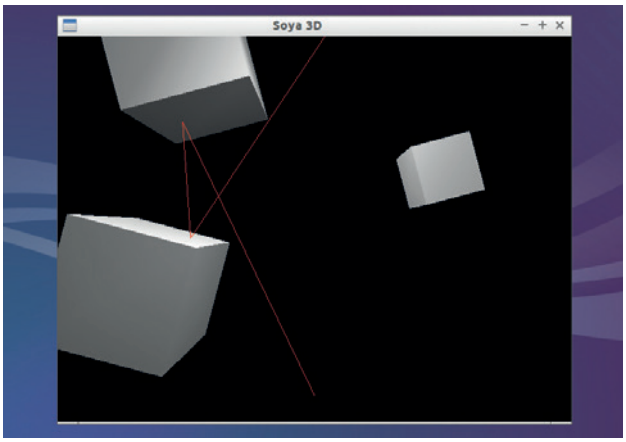
Soya 3D is a Python library that aims to free the programmer from this low-level detail while still getting the advantage of accelerated graphics. It lets you import models created in 3D modelling software (such as *Blender*) and manipulate them in your Python code.

Soya's stand-out feature is the number of really well commented tutorials that make it easy for anyone who knows Python to get started with 3D programming. You can get them from **http://home.gna.org/oomadness/en/soya3d/tutorials**. These take you through everything from creating a simple scene with just a single object in it, to creating complex animated scenes with interacting objects and varied lighting effects.

Accessible modelling

However, 'tutorials' is a bit of a misnomer. In reality, they're very well commented examples, but someone with basic knowledge of Python should find it quite easy to follow them and understand how *Soya* works. The only caveat with the tutorials is that they haven't been updated in quite some time, so they don't include all the latest features of *Soya 3*. However, they should include enough information to get you started and help you understand the principals behind the library.

If you want to get into 3D programming, *Soya* is a great place to start, because while it abstracts out a lot of the detail, it follows the same process as lower-level toolkits, so if you want to move on, you should find it quite easy.



Impress friends and colleagues with complex-looking 3D graphics – just don't tell them how easy they are to code.

PROJECT WEBSITE
www.lesfleursdunormal.fr/static/informatique/soya3d/index_en.html

Disk-saving GUI

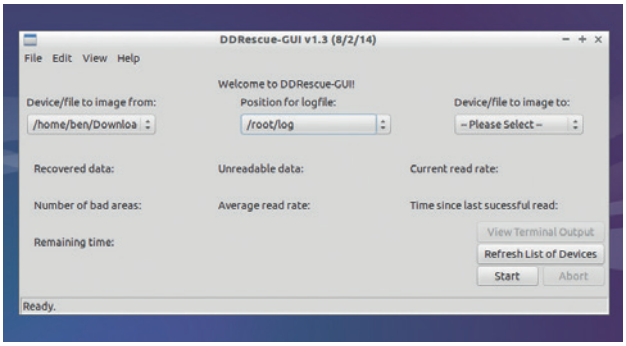
DDRescue-GUI

One of our most-used command line tools when it comes to manipulating disks is **dd**. This is a great tool for copying data to and from block devices such as hard drives and USB sticks. It's an old Unix tool, and the origins of its name have been lost in the mists of time (though some people claim it means 'data destroyer' because a misplaced argument can wipe all the data off a disk. This tendency to wipe data, provided the impetus for **ddrescue**, a newer program with more options to make it better suited to fetching data off a block device that's in the process of breaking. It enables you to specify the read order, and try particular blocks multiple times until you get a successful read. It's arguably the best tool for recovering data from a failing drive.

Unless you're very unlucky, you won't experience failing drives very often, and that means that you won't need **ddrescue** very often. The problem with command line tools that you don't use very often is that it can be hard to remember the correct options. In a situation where you've got a failing drive, missing an option can mean data is lost forever. *DDRescue-GUI* is a simple GUI wrapper for **ddrescue** that enables you to view and select the appropriate options, thus eliminating the risk of forgotten options.

As with all rescue tools, the worst time to learn it is when you need to use it, so it's worth trying out this software to make sure you're familiar with it should the worst happen. When a drive fails, it can deteriorate rapidly, and the longer you spend working out how to use the software, the more data you could lose. Once you've got the data off the broken disk, you can then spend as long as you need figuring out how to extract the information you need from the image file.

"The worst time to learn a rescue tool is when you need to use it."



As well as recovering data, you could use *DDRescue-GUI* as a graphical tool for writing an ISO image to a USB stick to make a live distro, or for backing up drives.

PROJECT WEBSITE
https://launchpad.net/ddrescue-gui

JavaScript framework

IO.js

Forks, spoons and sporks? Cutlery gone mad, or the spirit of democracy in open source software?

IO.js describes itself as a spork of Node.js, but that just adds more confusion, not less. A spork – for those of you unfamiliar with the term – is like a fork, but it's designed to supersede the original project so the two end up merging. This is in contrast to a true fork, which is designed to take the codebase in a new direction.

So that's a spork, but what's Node.js, and why would anyone want to spork it? Node.js is a JavaScript runtime that's designed for running event-driven, non-blocking applications. It's become particularly popular for building the back-ends for interactive web apps.

Node.js is open source, however, under the aegis of its current steward, an organisation called Joyent, development of Node.js appears to have stalled. Although there still appear to be people working on it, the pace of releases has slowed down.

For quite some time, Node.js has been based on an unsupported version of Google's V8 JavaScript engine. The people behind IO.js hope to increase the pace of releases, and they're already working on a more up-to-date version of V8 including support for the latest features in EMCA Script 6 (ES6). Unlike Node.js, IO.js isn't run by a company, but by developers from the community. The hope is that this will mean the project will not be dominated by a single party.

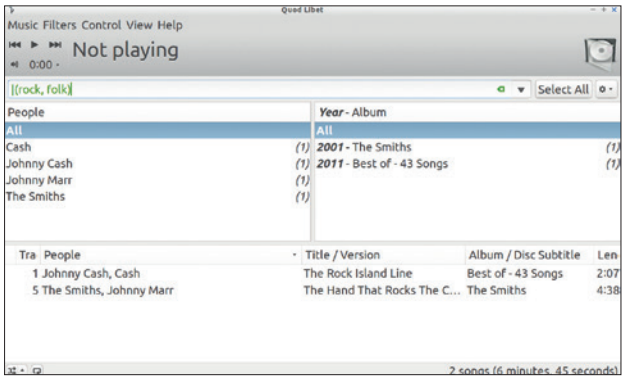
IO.js is compatible with the Node Package Manager, so many projects should work on both Node.js and IO.js at the moment, although this may change if the two projects diverge further. It may be slightly confusing to some new users that Node.js (the older project) is on version 0.10, while IO.js has just released version 1.0. This isn't intended to signify that IO.js is more mature; they're just using a different release numbering system and using 1.0 to show that they're a fully independent project.

"The people behind IO.js hope to increase the pace of releases."

PROJECT WEBSITE
https://iojs.org

Music manager

Quod Libet



Powerful functionality, not graphical frivolity, is what makes *Quod Libet* stand out.

There are some really good music players for Linux, so in order to stand out a player has to do something special. In *Quod Libet*'s case, that something special is its search feature. It allows you to search tags, and also combine different tags, negate particular search options and others. Basically, it enables you to very easily specify exactly what music you want on your playlist.

For example, do you want all rock music, but want to keep it short and snappy? Well, you can search for all songs that have the tag rock, but are less that five minutes long. How about all pop music released before 1979? No problem. This is all done through *Quod Libet*'s text-based search language, detailed at **https://quodlibet.readthedocs.org/en/latest/guide/searching.html**

Quod Libet works on the principal that you know wha you want to listen to, so gives you the power to specify it. This is in direct contrast to some other music players that try to guess what you want and make the decision for you.

Of course, all this requires you to have a good source of properly tagged music, otherwise *Quod Libet* will struggle to find the songs you specify. *Quod Libet* does come with some tools to help you tag you files, but it doesn't support automatic tagging. Provided all your songs are well tagged, *Quod Libet* will easily scale to collections with large numbers of songs. In fact, its powerful search features are probably best suited to albums with thousands of songs or more.

The latest version (3.3) comes with a few new features including better handling of diacritic marks in searches, easier access to song lyrics (via **lyrics.wikia.com**) and support for more file formats.

If you've been struggling to manage a large audio collection on Linux, and you're comfortable using text-based search tools, *Quod Libet* could be the music player you've been waiting for.

PROJECT WEBSITE
https://code.google.com/p/quodlibet

Image compressor

Mozjpeg

Images: without them, the web would be a boring place. Whether they're cute kittens, flashy logos, or textured backgrounds, images are what change web page from piles of text into an enjoyable medium. However, they take up bandwidth, and that means they slow down page loads, and cost more money.

You can – and should – compress images. However, the most common form of image compression is JPEG, and that's over 20 years old. There are better options, but the nasty world of patent enforcement means they're rarely used.

Mozjpeg isn't a new compression standard, but an optimised JPEG compression tool using newer techniques behind the scenes to squeeze images down to the smallest possible size while retaining as much detail as possible. The gains aren't huge (around 5–10%), but if images take up a significant portion of your web traffic, this could be a meaningful difference. After all, a 5% reduction in bandwidth bills and storage requirements could result in a huge saving for popular websites. Similarly, a 5% speedup in page load times will result in a better experience for your users.

The images from *Mozjpeg* are completely compatible with normal JPEG decoders, so you can view them in older web browsers without any plugins or additional codecs.

Mozilla, the creator of the tool, is keen to point out that *Mozjpeg* is specifically tuned for web images, and isn't intended as a general purpose JPEG tool. If you're planning on printing images, then you should stick with a standard JPEG library, as this will give better images at this quality.

"Mozjpeg is an optimised JPEG compression tool for the web."

One of these images is compressed 9% more than the other. Can you tell which one? We can't, but we know that one loads faster.

PROJECT WEBSITE
https://github.com/mozilla/mozjpeg

Graphical command tool

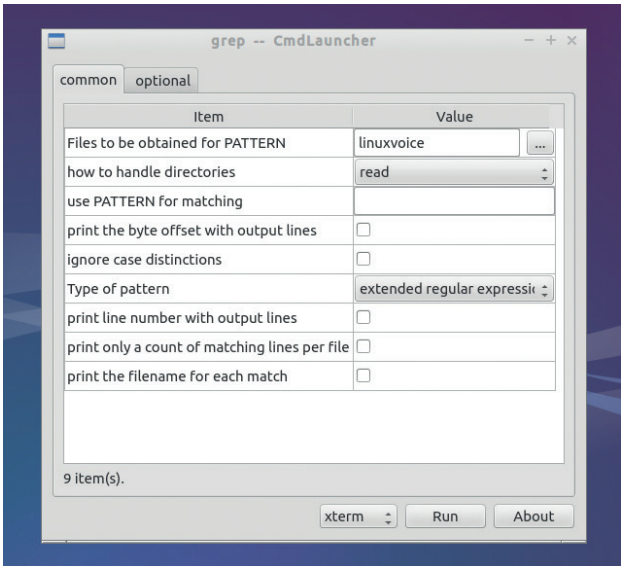
CmdLauncher

The Unix shell interface is an amazingly powerful thing. It's arguably the most powerful interface ever created for computing. However, it can be intimidating for new users. *CmdLauncher* is a GUI for command line tools to allow new users to use things like **grep** without delving down into the text interface.

CmdLauncher isn't built for a specific set of commands; instead it uses configuration files created for each command. These are fairly straightforward and can easily be created for new commands. It only comes with two: **grep** and **upx** (a tool for creating compressed executable files).

Another way of looking at *CmdLauncher* is as a GUI builder for text-based tools. If you need to help someone use some particular arcane command line tool, you can quickly create a config file that means they can use the tool in a friendly window without having to know what's going on underneath. By deciding on the particular options you make available, and how you name them, you can help them use particular tasks. You could even use it to create an interface for any shell scripts you create. If you do create useful config files, be sure to share them as it would be helpful if the software came with more than the two examples.

CmdLauncher can only be used for individual commands, and you can't set up chains of *CmdLauncher* instances that each pipe data into the next, so it's not a complete replacement for the shell interface. Advanced users will always have to run the command line, but this is just powerful enough to help some people who aren't interested in computing get things done without making the plunge to the shell.



Never before has it been so easy to get started with **grep**.

PROJECT WEBSITE
http://cmdlauncher.topbug.net

FOSSPICKS Brain Relaxers

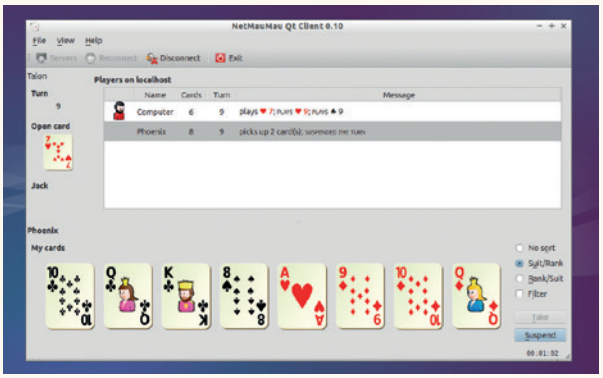
Online card game

NetMauMau

The card game Mau Mau is similar to the game Uno: you have to get rid of all your cards by playing a card of the same suit or number as the one previously played. A few cards have special rules, but there doesn't seem to be a general consensus on which cards have what special rules. Instead, it's common for different groups of people to play by slightly different rules. The makers of *NetMauMau* outline their rules on the GitHub page (**https://github.com/velnias75/NetMauMau**) and far be it from us to claim these are wrong. If you disagree, there are a few command line options to change the rules to something that suits you better.

NetMauMau is a client–server based game that you can play either with other players across the internet, or with a computer AI on the network, and there can be up to five people in each game.

We didn't find the AI especially challenging, but we should point out that the reviewer spent many nights perfecting his Mau Mau skills in Tanzania where he didn't have a TV or computer, so had nothing else to do apart from create increasingly complex techniques for shedding cards in Last Cardi (as Mau Mau is known in Swahili). The trick, as with so many games, is to always think not of what you can play now, but what you'll want to play in the future. There's more to this game than it initially seems. The one thing missing from



The internet has a full list of optional rules, though few of them are implemented in *NetMauMau*.

NetMauMau is any way to search for public servers. If you fancy a game, you're limited to either playing against AI, or locating opponents for yourself. Perhaps this is for the best, as our memory of Mau Mau is as a social game, and that would be lost on people you didn't know.

PROJECT WEBSITE
nhttps://github.com/velnias75/NetMauMau

Racing game

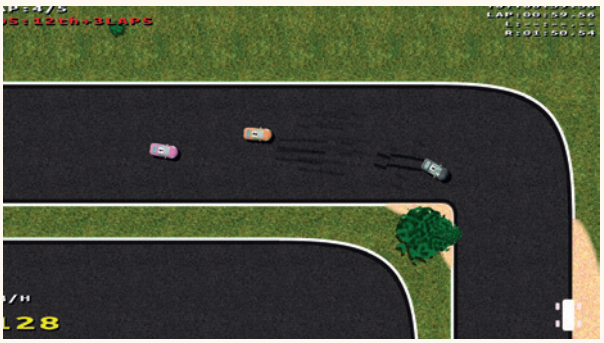
Dust Racing 2D

Car racing games are hard to do well. To feel right, they need to have smooth graphics, physics that work properly, and AI players that present the right levels of difficulty. *Dust Racing 2D* is a top-down car racing game that gets all these right.

It sometimes feels a little like you're driving on an oil slick, and learning to handle the car is a key part of advancing in the game. It claims to be a 2D game, but this isn't quite right. Although it doesn't have full 3D, it does have some effects that gives a 3D feel to the top-down view. Our test system has Intel HD graphics and no graphics card, and on this it ran smoothly and looked great, so you don't need a top-spec graphics card or proprietary blobs to play.

The controls are simple (by default just the arrow keys), so it's quick to get started, even for people not used to racing games. There are five tracks to play by default (although only one is unlocked until you prove your racing chops), or you can design your own with the level editor.

Overall, *Dust Racing 2D* isn't a complex game, but it is great fun to slide around corners and bounce off your competitors in a mad dash



Car control isn't our strong point, but what we lack in finesse, we more than make up pure speed.

to the finish. There are various two-player options (regular race, duel and time trials), so you can challenge your mates to some open-source racing fun.

"It's great fun to slide round corners and bounce off your competitors."

PROJECT WEBSITE
http://sourceforge.net/projects/dustrac/